CLAIMS

1. A method for a communications network, comprising the step of receiving a program code, comprising a plurality of instructions for the communications network, **characterized by** the steps of

- dividing the program code into a plurality of sequences (7),
- defining, based on the program code, a plurality of relocation objects (10), each corresponding to a dependency relationship between two or more of the sequences (7), and
- allocating the sequences (7) to a processor instruction memory (4).

2. A method according to claim 1, comprising the steps of forming at least one directed graph, based on at least some of the sequences (7) and at least some of the relocation objects (10), and determining a longest execution path through the directed graph.

3. A method according to claim 2, comprising the step of entering at least one state preserving operation (NOP) in the instruction memory (4), so as to make at least two execution paths equally long.

4. A method according to claim 3, comprising the step of moving at least one sequence in the instruction memory (4).

5. A method according to claim 3 or 4, wherein the length of the at least two execution paths correspond to the longest execution path.

6. A method according to any of the preceding claims, comprising the step of determining the existence of any circle reference by any of the relocation objects (10) between any of the sequences (7).

7. A method according to any of the preceding claims, comprising the step of linking at least one sequence, obtained by the step of dividing the program code, to a sequence, obtained by dividing another program code.

8. A processing system for a communications network, comprising an assembler adapted to receive a program code, comprising a plurality of instructions for the communications network, **characterized by** the assembler being adapted to divide the program code into a plurality of sequences (7), and define, based on the program code, a plurality of relocation objects (10), each corresponding to a dependency relationship between two or more of the sequences (7), and a linker being adapted to allocate the sequences (7) to a processor instruction memory (4).

9. A processing system according to claim 8, wherein the assembler is adapted to form at least one directed graph, based on at least some of the sequences (7) and at least some of the relocation objects (10), and the linker is adapted to determine a longest execution path through the directed graph.

10. A processing system according to claim 9, wherein the linker is adapted to enter at least one state preserving operation (NOP) in the instruction memory (4), so as to make at least two execution paths equally long.

11. A processing system according to claim 10, wherein the linker is adapted to move at least one sequence in the instruction memory (4).

12. A processing system according to claim 10 or 11, wherein the length of the at least two execution paths correspond to the longest execution path.

13. A processing system according to any of the claims 8 to 12, wherein the linker is adapted to determine the existence of any circle reference by any of the relocation objects (10) between any of the sequences (7).

14. A processing system according to any of the claims 8 to 13, wherein the linker is adapted to link at least one sequence, obtained by dividing the program code, to a sequence, obtained by dividing another program code.